Please replace the paragraph beginning at page 1, lines 7-11, with the following rewritten paragraph:

--Background of the invention

In the field of data and computer communications there is an increasing need for high speed/high bandwidth products. Prior art references relating to packet switching and more specifically to data stream decoding and pertinent to the present invention include:--

Please replace the paragraph beginning at page 1, lines 32-39, with the following rewritten paragraph:

--The invention relates to a device for data stream analyzing. Said device is able to recognize different data streams and then start other processors or functionalities to store or check data in a data stream. Special features are: a compare processor, a compare instruction memory, a data stream pipeline, a multiplexer and a multiplexer control unit, making it possible to test packet data under program control using several instructions and under several clock cycles even though said data is moving forward in the pipeline and even though other bytes of data are entering the device.--

Please replace the paragraph beginning at page 2, line 6, with the following rewritten paragraph:

--Fig. 2 is a block diagram of the multiplexer control unit of Fig. 1.--

Please replace the paragraph beginning at page 2, line 9, with the following rewritten paragraph:

--Detailed description of the invention--

Please replace the paragraph beginning at page 2, lines 10-12, with the following rewritten paragraph:

--The invention is preferably implemented as an integrated circuit (IC) having an electrical interface to the outside. The invention comprises a number of physical or logical units as illustrated in Fig. 1, including:--

Please replace the paragraph beginning at page 2, line 35-page 3, line 2, with the following rewritten paragraph:

--When the compare processor 5 has come to such kind of conclusion it might want to report something to a result field or an option field, see below. This is done by starting up a save sequence. A start address for a save sequence will be sent from the compare processor 5 to the save engine 6. Said save engine 6 examines the incoming address and decides if it is a save regarding the result field or the option field. According to this decision the address is placed in either a bit save fifo register 61 or a stream save fifo register 62 respectively.--

Please replace the paragraph beginning at page 3, lines 4-5, with the following rewritten paragraph:

--The bit save unit 7 has three functions: it can set bits in the result field, perform checksum control and length control.--

Please replace the paragraph beginning at page 3, lines 10-26, with the following rewritten paragraph:

--The delayline 1 preferably comprises a 23 shifts deep, 1 byte wide shift register. The 16 first positions of the shift register are reachable from the compare processor 5 through a multiplexer 2. The two last positions are connected to the two save units 7,9 (bit save and stream save). The stream save unit 9 is actually only using the very last position, and only the bit save unit 7 needs the last two positions because the checksum control works with 16 bits at a time. There are five positions that are prevented from being accessed by the parsing function of the compare processor 5 and by the save units 7, 9 (bit save and stream save). The reason for a delay before the byte stream arrives to the save units 7, 9 is that all start addresses sent from the compare processor 5 to the save units 7, 9 are queued in a fifo register. Depending on how many save sequences in the queue and how long they are, this might in some extreme situations generate an error. This is because vital data already have passed through the delayline before a save sequence is started. The actual delay needed to secure that no such error occurs is 4*64=192 clock cycles. 64 is the maximum length of a save sequence and 4 is the maximum of start addresses waiting to be executed. However, simulations have shown that five delay cycles are enough, since all save sequences normally written are very short--

Please replace the paragraph beginning at page 3, line 28-37, with the following rewritten paragraph:

--A characteristic function of the invention is that it automatically keeps track of where a specific byte has its location in the delayline. The programmer only needs to specify which tag, i.e. which number the byte has, where the first byte in a packet is number zero, the second is number 1 and so on. This is why every byte arriving to the delayline 1 should be tagged (numbered). The tagging operating could easily be done by just adding an extra field in every shift in the delayline 1 inheriting the byte's tag. But this is disadvantageous in two aspects. First, much silicon would be used to implement the extra field in the delayline 1. Second, when the parser wants to look at a specific tag it would take a lot of time if every shift had to be searched to find the wanted tag.--

Please replace the paragraph beginning at page 4, lines 1-10, with the following rewritten paragraph:

--Instead, the present invention has solved the above problem by making a part of the delayline multiplexable; said multiplexable part of the delayline preferably includes the 16 latest incoming bytes. Worst case for the length of a packet is 1 byte (erroneous), but since the first 12 bytes always contain the OSI Media Access Control address (MAC-address), no useful information can be extracted if the packet is shorter than 13 bytes. These packets will force the compare

processor 5 to begin with the next packet at once and their DV (data valid) signal will be unset so the rest of the device or a switch will never see it. With a limit of at least two clock cycles (bytes) between different packets it is possible to guarantee that never more than two packets exist at the same time in the delayline 1.--

Please replace the paragraph beginning at page 4, lines 12-16, with the following rewritten paragraph:

--According to the ethernet standard the IFG (Inter Frame Gap), which means the distance between packets, is at least 20 cycles, but a smaller distance is always desirable. E.g., a minumum distance of 6 cycles makes it possible to easily extend the device to be able to take care of SONET frames (An alternative ISO-028 Layer 2 frame instead of ethernet.--

Please replace the paragraph beginning at page 4, lines 18-28, with the following rewritten paragraph:

--Fig. 2 illustrates that the multiplex control unit 3 uses two identical Tag Units 32, 33 (TU), one for each possible packet, a Controlling Statemachine 31 (CS) to control the TU:s 32, 33 and a TU multiplexer 34 to choose which one of the TU:s 32, 33 that the compare processor 5 is interested in. One TU includes a tagfield register 321, and a lastfield register 322, some adders and a simple statemachine 323. The other TU 33 is identical. When a packet arrives, the tagfield register 321 starts to increment for every byte.

a9

When the DV signal becomes false again the tagfield register 321 stops counting and the lastfield register 322 starts to increment. The TU 32 sends an "end_of_packet" signal when the lastfield register 322 reaches the number of shifts in the delayline 1. If the packet was shorter than 13 ~~bytes a~~ "~~too_short~~" signal will be generated.--

Please replace the paragraph beginning at page 5, lines 11-15, with the following rewritten paragraph:

a10

--Referring again to Fig. 1, a feature of the device according to the present invention is that the compare processor 5 and the compare instruction memory 4 together act as a programmable parser. The description of the full instruction set of said parser is not set forth herein, but some instruction types are mentioned below. The parser uses four registers 51, 52, 54, 55 to fulfil its tasks.--

Please replace the paragraph beginning at page 5, lines 28-35, with the following rewritten paragraph:

a11

--All instructions are executed in one clockcycle, except in two cases. This is possible because the compare processor unit receives two instructions every clockcycle from the compare instruction memory 4 which is of the double ported memory type. This feature decreases the total amount of clock cycles needed for the compare processor 5 to parse a packet, thereby decreasing the needed size of the delayline. Some instructions are able to start

a11

save sequences. Said instructions have a field that tells what address in the save instruction memory 8 shall start the execution. Save address 0x00 will not generate a start of a save sequence.--

Please replace the paragraph beginning at page 6, lines 6-14, with the following rewritten paragraph:

a12

--The save engine 6 takes the address sent from the compare processor 5 and determines if it is the start address of a bit save sequence or a byte stream sequence. Thereafter the address together with the current value of the base register 54 is put in the specific fifo 61, 62. The value of the base register 54 is needed for all save instructions that are using tag numbers. When the device according to the invention is programmed, a constant is written to the save engine 6 to tell where bit save sequences end in the save instruction memory 8. This feature exists because it is hart to tell how many instructions are needed to the different parts and it is more expensive to map two memories than one twice as big.—

Please replace the paragraph beginning at page 6, lines 21-26, with the following rewritten paragraph:

a13

--The checksum unit 73 executes a checksum control instruction which performs a 16-bit one complement addition. The unit needs to know what tag to start the execution from (Tag) and how many

bytes the checksum should cover (Length). If there are checksum errors (i.e. the sum differs from 0xFFFF) the unit writes to the result field 76. Further, this block needs the value of the base register 54 as it was when the compare processor 5 sent the start address of the current save sequence.--

a13

Please replace the paragraph beginning at page 6, lines 29-36, with the following rewritten paragraph:

--The bit unit 74 executes a bit save command which bitwise "xor"-ise one selected byte in the result field 76 with the data field. In other words, all bits which are set in the data field will invert the corresponding bit in the result field 76. It is only possible to invert one specific bit one time per packet, this is because e.g. an OSI Layer 3 error could be found in many ways, but if the bit which indicates a Layer 3 error is set an even number of times, this would look like a correct Layer 3 packet in the result field 76. The address field indicates which byte, of three possible bytes in the result field 76, should be written to.--

Please replace the paragraph beginning at page 7, lines 1-12, with the following rewritten paragraph:

a14

--The length error unit 75 is the most complex unit and investigates lengths in a packet and is used with one or more length control instructions. In a network there might occur packets that have been cut off. This causes many kinds of errors, e.g. if layer 4 is shorter

than two bytes the result field 76 should indicate Layer 3 error but not Layer 2 error. The length error unit 75 consists of two identical checkboxes and one controller. A checkbox needs to know at which tag to start the measurement from, what kind of comparison it is supposed to perform (more, less, equal or not equal) and what length to match this comparison to. If a checkbox detects a length error, a field which is part of the instruction indicates which bit, of four possible bits in the results field 76, should be written to. As with the checksum unit 73, this unit 75 also needs the value from the base register 54 as it was when the compare processor 5 sent the start address of the current save sequence.--

Please replace the paragraph beginning at page 7, lines 21-28, with the following rewritten paragraph:

--The electrical interface of a preferred embodiment of the invention to the outside world is described in conjunction with fig. 3. It includes an input interface and an output interface. The input interface of the invention includes nine input terminals for a synchronous, eight bit wide, serial data stream, and a data valid (DV) signal, both used by the data that should be decoded. The input interface also includes a programming interface that comprises an 8-bit address bus, and 18-bit data bus, a chip select and a write enable signal for programming the two instructions.

a15

These 28 input terminals are used to program the invention after power on.--

Please replace the paragraph beginning at page 7, line 37-page 8, line 7, with the following rewritten paragraph:

a16

--A typical application for the present invention is for packet switching in a computer network together with a packet switch by extracting information, especially addresses, from the packet headers, because it is possible to test data using several instructions and under several clock cycles even though said data is moving forward in the delayline (1) and even though other bytes of data are entering the device. One of the features of the invention is that the decoding of the protocol is programmable. This is a major advantage because new or different types of protocols can be handled by just reprogramming the device. There will be no need for changing the hardware. This could save time and money for companies responsible for providing, maintaining and updating network switches.--

In the Claims.

Please replace the centrally-located one-word paragraph at page 9, line 1, with the following rewritten left-flush paragraph:

--What Is Claimed is:--.

Please amend Claims 1-8 by substituting the following amended claims for the pending claims having the same number: